# Deep Space Network Scheduling Using Evolutionary Computational Methods

Alexandre Guillaume[1], Seugnwon Lee, Yeou-Fang Wang, Hua Zheng, Robert Hovden, Savio Chau, Yu-Wen Tung, Richard J. Terrile
Jet Propulsion Laboratory, 4800 Oak Grove drive, 91109-8099 Pasadena CA
[1]alexandre.guillaume@jpl.nasa.gov , tel: 818-393-6899

*Abstract*—The Deep Space Network (DSN) is an international network of antennas that supports all of NASA's deep space missions. With the increasing demand of tracking time, DSN is highly over-subscribed. Therefore, the allocation of the DSN resources should be optimally scheduled to satisfy the requirements of as many missions as possible. Currently, the DSN schedules are manually and iteratively generated through several meetings to resolve conflicts. In an attempt to ease the burden of the DSN scheduling task, we have applied evolutionary computational techniques to the DSN scheduling problem. These methods provide a decision support system by automatically generating a population of optimized schedules under varying conflict conditions. These schedules are used to decide the simplest path to resolve conflicts as new scheduled items are added or changed along the scheduled 26 weeks.[12]

This paper presents the specific approach taken to formulate the problem in terms of gene encoding, fitness function, and genetic operations. The genome is encoded such that a subset of the scheduling constraints is automatically satisfied. Several fitness functions are formulated to emphasize different aspects of the scheduling problem. The optimal solutions of the different fitness functions demonstrate the trade-off of the scheduling problem and provide insight into a conflict resolution process.

TABLE OF CONTENTS

1. INTRODUCTION.....1
2. CURRENT DSN SCHEDULING PROCEDURE .....1
3. OPTIMIZATION WITH A GENETIC ALGORITHM.....2
4. RESULTS .....3
5. FUTURE PROSPECTS AND IMPROVEMENTS .....4
6. CONCLUSION .....5
REFERENCES .....5
BIOGRAPHY .....5

## 1. INTRODUCTION

The Deep Space Network (DSN) is a collection of radio antennas and their support hardware, software, and personnel. Its primary function is to connect spacecraft with their controllers at JPL and JPL's partners. The DSN network consists of three communication facilities spaced by 120 degrees (in longitude) from each other: the Goldstone Deep Space Communications Complex in California, the Madrid Deep Space Communications Complex in Spain and the Camberra Deep Space Communications Complex in Australia. This strategic placement permits constant observation of spacecraft as the Earth rotates. There are over 16 antennas in DSN to support about 30 missions. Allocating these resources to support the missions is a basic DSN scheduling problem. There are many constraints to this scheduling problem. For example, a mission can only be seen during certain period of time at each antenna. We call this period of time view period. This is related to the location of the antenna and spacecraft trajectory. There are other types of constraints such as equipment conflict, radio frequency interference, and work crew availability. Besides single resource for single mission, DSN also supports single resource for multiple spacecraft and multiple resources for single mission. Roughly speaking, there are about 500 schedule items in a week. DSN normally performs hundreds of changes for each week. DSN scheduling is a dynamic and continuous environment where changes constantly occur [1][2]. Everything from calculating where to point the antennas to the spacecraft tracking operations starts with an official DSN schedule. Mission operations cannot be successful without appropriate time reservations in the schedule. With increasing demand of tracking support from missions, DSN is currently over-subscribed. In order to make the best use of the DSN resources, optimally allocating resources becomes an important issue. We propose an evolutionary computing approach to use Genetic Algorithm (GA) to optimize the DSN schedule. Based on the simulation results, the GA approach we are taking can reduce overall number of schedule conflicts and produce a more efficient schedule.

## 2. CURRENT DSN SCHEDULING PROCEDURE

DSN scheduling involves multiple users with various types of communication needs. This includes ground-based science activities and spacecraft tracking. Each user and activity has its unique needs and requirements. There are three steps in the DSN planning: long-range planning, mid-range scheduling, and near-real-time scheduling. Long-range planning is from 1 year to more than 10 years from now. In this timeframe, DSN forecasts future loads of the network and determines the supportability level for each mission. Mid-range scheduling is from 8 weeks to 1 year. In this stage, initial schedule is generated based on the user requirements negotiated in the long-range planning. Then detailed negotiations start to eliminate conflicts in the schedule. DSN schedulers coordinate the change activities and resolve DSN conflicts. Project/mission schedulers are

1

[1] 1-4244-0525-4/07/$20.00 ©2007 IEEE
[2] IEEEAC paper #1210, Version 2, Updated December 6, 2006

working with each other to come up with a solution proposal. Through proposal exchange and meetings, a virtually conflict-free schedule is generated toward the end of this process interval. Near-real-time scheduling then starts around 8-week boundary. It is within this time frame that schedules are prepared to be ready for real-time operation. DSN schedulers continue to negotiate with project schedulers and handle contingencies through out the process toward real-time.

There are two major scheduling issues in the mid-range and near-real-time scheduling area: one is to generate an initial schedule and the other one is to resolve conflicts when they occur. In the current process, they are both labor-intensive. Generating a feasible schedule is difficult. Generating an optimal schedule is almost impossible. There is a large number of combinations for allocations in a normal case (such as 500 tracks in 16 antennas for a week). Human can only try out certain amount of solutions based on her or his experience. A normal computer may take months if not years to explore all scheduling possibilities. Therefore an optimization algorithm with reasonable turn around time is preferred in this situation.

## 3. OPTIMIZATION WITH A GENETIC ALGORITHM

When all the requirements have been centralized, the establishment of the schedule is equivalent to an optimization problem with multiple constraints and objectives. There are several efficient numerical techniques to solve these kinds of problems: genetic algorithms, simulated annealing, hill climbing, *etc*… Several reasons motivated our choice of technique. First, over the years, genetic algorithms have been used with great success to explore complex search spaces and have demonstrated human competitive improvement over conventional methods [3]. Secondly, this technique is well suited to solve scheduling problems [4]. Finally, we have both the expertise in genetic algorithms and the knowledge of the DSN operation. In this work, we present an evolutionary approach to the scheduling problem of the Deep Space Network.

We first construct the genome with the essential characteristics of the forecast week. Each task is characterized by an identification number, a duration, a set up and tear down period and schedulable intervals (view periods). The algorithm looks for candidate solutions (schedules) among the different view periods. In this case, the gene is simply an integer number (in minute) chosen randomly between zero and the sum of all view periods. A tracking number is assigned to each period so that when a random number is chosen within the interval, it can be traced back to a given schedulable interval. In other words, this tracking number assures the mapping between the phenotype and the genotype. There are typically about 500 tasks. As a first step towards establishing an efficient scheduling tool for the Deep Space Network, we only consider single events, e.g. disregard events involving several, either, antennas (antenna arraying) or spacecrafts (Multiple Spacecraft Per Antenna, MSPA). There are usually 140 of those normal tasks that also possess schedulable intervals. Each chromosome has therefore around 140 genes. There are usually up to 50 different schedulable intervals for each of these tasks.

Before further discussing the specifics of the genetic algorithm, we describe the implementation details as the optimization of the algorithm was greatly influenced by it. In order to use the existing infrastructure, we use a dedicated web service located at JPL. The public interface to this service is achieved through a WSDL, or Web Services Description Languages. The SOAP protocol is used to exchange XML files between the users and the service. The genetic algorithm is coded in C/C++. We used the gSOAP toolkit to bind the XML data to a C++ structure. So the genetic algorithm run on a client machine that make conflict calls to the server at each iteration. Using the DSN framework has the advantage that our code could readily be inserted into the current DSN schedule tools. However, having the computation and the conflict check performed at two different points introduce a serious overhead as approximately two thirds of each iteration is spent in communication.

Each solution should fulfill numerous scheduling objectives: 1) maximize the number of tasks without conflicts, 2) maximize the mission coverage, 3) minimize the number of conflicts, 4) minimize schedule items overlaps. We initially built different fitness functions for each objective. We then implemented a multi-objective approach by simply building a fitness function as a linear combination of different objectives. This allows us to fine tune the fitness function by adjusting the relative weights of each component. The number of tasks without conflict is obtained by subtracting the number of tasks with conflicts from the number of tasks in the corresponding schedule. The total mission coverage is obtained by adding the duration of each task without conflict. The number of conflicts is given by the sum of the number of conflicts from each task. Note that a task can have more than one conflict so the number of conflicts is different from the number of tasks with conflicts. Overlaps are determined by the number of so-called facility conflicts which indicates that more than one mission are assigned to the same antenna/facility in some time period.

The program starts by requesting the current forecast week, which is the schedule for the coming week that has been optimized by the DSN scheduling team. Most of the time, the forecast week is therefore conflict free. The performance of our code can be simply evaluated by comparing our solution to this original solution. The initial population is chosen, as is customary in genetic algorithms, by randomly picking the genes in the pool of possible solutions (schedulable intervals).

We also implemented an evolution strategy for the mutation step. Each gene starts with a different mutation step size which is determined by 10 percent of the range of the gene. As the number of generations increases, the size of the mutation step decreases linearly We introduced this scheme in order to explore the search space efficiently at the beginning of the evolution process and to fine-tune the search step as the solution gets close to the ideal solution.

## 4. RESULTS

We present the results obtained from the forecast for the second week of October, 2006. There were 412 tasks for this week among which 153 were normal with schedulable intervals. The fitness of the initial schedule was evaluated before the initialization of the genetic algorithm. There were 343 tasks without conflict, 10.5 week mission coverage, 706 conflicts and 0 facility conflicts. These values are indicated with a red horizontal line as a reference on Figure 1 to Figure 4 (which correspond to objective 1) to 4) respectively):

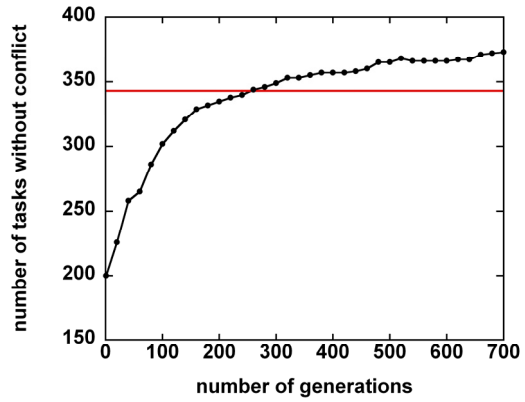


**Figure 1: Evolution of objective 1) with the number of generations. The red line indicates the fitness of the initial schedule, equal to 343 tasks without conflict.**

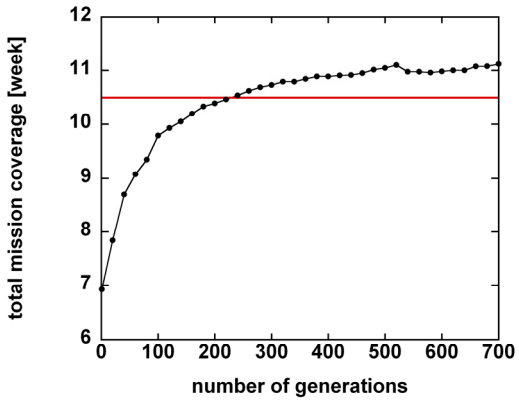


**Figure 2: Evolution of objective 2) with the number of generations. The red line indicates the fitness of the initial schedule, equal to 10.5 week.**

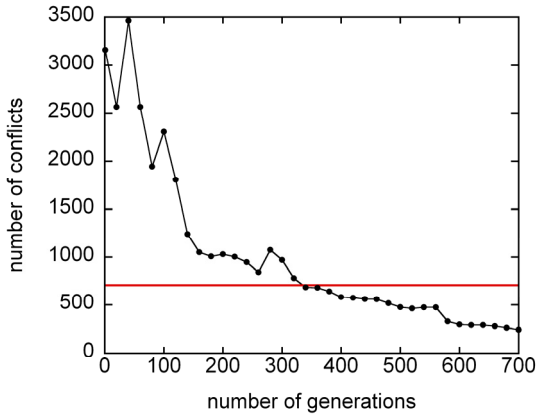


**Figure 3: Evolution of objective 3) with the number of generations. The red line indicates the fitness of the initial schedule, equal to 706 conflicts.**

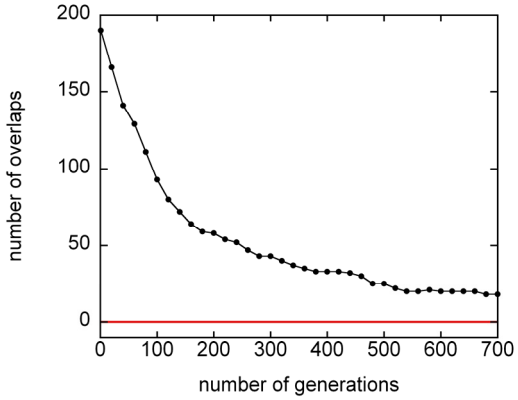


**Figure 4: Evolution of objective 4) with the number of generations. The red line indicates the fitness of the initial schedule, equal to 0 overlap.**

For this optimization problem, we have made a single objective out of the four objectives listed in the section 3 with the relative weights of the four objectives being one 1 for objective 1), 2 for objective 2), 1 for objective 3) and 8 for objective 4). We chose these weights for different reasons. First, we discovered over the course of our study that minimizing the number of overlaps was the hardest objective to fulfill which explains the relatively high weight. Then, we gave the remaining weights equal importance as an initial guess. However, objective 1) and 3) are calculated with *partly* common information, which explains why the weight of objective 2) is equal to the sum of their weights. We used genetic algorithm parameters as follows. The population size was 500, the replaced population size for each generation 300, the mutation rate 0.02, the crossover probability 0.7 and the maximum number of generation 500 . We used a binary tournament parent selection scheme with 0.9 probability and a two-point crossover for the recombination operator. We chose a

Gaussian mutation operator with a mutation step that increased linearly with the number of generation, starting at 10% of the upper bound defining a gene.

The genetic algorithm clearly outperforms the initial schedule for three of the four objectives, as seen on Figure 1 to Figure 3. However, even though the relative weight for the minimization of the number of overlaps is eight times higher than the other objectives, the number of overlaps is still higher (18) than the initial solution (0) after 700 generations (see Figure 4). It is also worth noticing that one overlap between a given task *i* and task *j* appears as two overlaps in our count (task *i* with task *j*, and task *j* with task *i*). So, there are really 9 facility conflicts in the solution at the 700th generation. Moreover, the program really operates according to the few rules we described. In reality, exceptions to these rules are used for human operations to permit some flexibility and resolve conflicts. For instance, some artificial tasks may be used by schedulers to fill in the gaps and thus minimize the number of gaps between tasks.

We allowed the program to restart after 10 generations if no progress was made. To confirm the utility and efficiency of this process, we computed the diversity of the population for different generations. There is no single measure for the diversity. For a given generation, we calculated the distance *D* defined as followed:

$$D = \frac{1}{N_{pop} \times N_g} \sum_{p}^{N_{pop}} \sqrt{\sum_{j=1}^{N_g} \left( x_{p,j} - \bar{x}_j \right)^2} \,, \qquad \textbf{(1)}$$

where $x_{p,i}$ is the gene *i* of the individual *p*. The diversity is therefore expressed here in minutes. $N_{pop}$ represents the number of individuals in a population and $N_g$ the number of genes. We implemented this change on a subsequent run with the same initial forecast week but with different relative weights (1, 2, 1, 16 respectively). The results were somewhat comparable for the first three objectives but worst for objective 4). The number of tasks without conflict is shown along with the diversity on Figure 5**Error! Reference source not found.**. We chose a higher mutation probability of 0.05 for the restart process. As indicated by the finite slope after the first three spikes, the restart process is indeed efficient at improving the objective further. Moreover, for this specific objective, 1), the two first restarts allow the algorithm to find a better solution than the initial forecast week (indicated by a red line on **Error! Reference source not found.**). Logically, we found with other experiments that the height of the peak indicating a restart, was proportional to the restart probability.

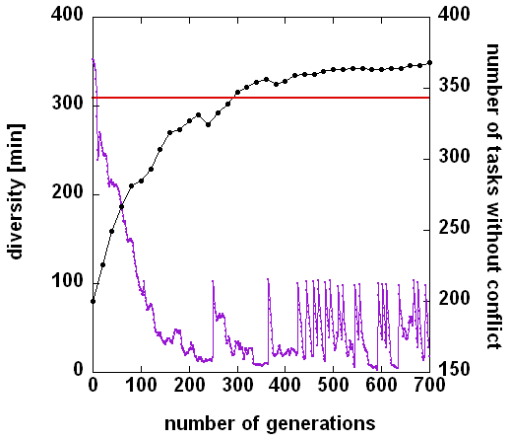


**Figure 5: Diversity (purple curve) and number of tasks without conflict (black curve) versus the number of generations. The red line indicates the fitness of the initial schedule, equal to 343 tasks without conflict.**

## 5. FUTURE PROSPECTS AND IMPROVEMENTS

The results we show on Figure 1 to Figure 4, were the best we obtained with this method for a single objective. Beside the variation of the different parameters to optimize the program, we tested our method with different forecast weeks. To our surprise, we observed a spread in performances. Most of the time, the program could reach and beat the initial solution of the forecast week for the three objectives 1), 2) and 3). However, some other times, the program did not reach any of the four objectives calculated for the initial solution. More than the difference in missions from one week to the other, we think this fact emphasizes the importance of other objectives and constraints, and the need for human input to resolve in a non-systematic fashion some hard conflicts.

There are many improvements possible. Since the time to perform a simulation was mostly spent in communication between the client and the server, a substantial amount of time (about two thirds) could be gained by moving the conflict check on the same machine that is running the program. For this work we used a mutation step that was calculated with respect to the upper bound of the gene. However, it is possible to achieve the same upper bound of the genes, *e.g.* the same sum of all schedulable intervals, with two different series of schedulable intervals. The sum of a few intervals can be equal to the sum of more, but smaller intervals. A percentage doesn't capture this possibility. Since this program is supposed to, at least, mimic the current scheduling process, and at most propose alternative solutions, it would be logical to include most of the constraints and objectives currently used.

We currently convert a multiple objective problem to a single objective problem by weighting the multiple objectives. The multiple objectives often compete with each

other, which means that improving one objective is accompanied by degrading other objectives. In such a situation, the optimization problem leads to multiple optimal solutions that demonstrate a trade-off between different objectives rather than one single best solution that optimizes all the objectives. Several algorithms are available for the multi-objective optimization problem and can be easily implemented into our evolutionary computing framework.

## 6. CONCLUSION

We proposed an evolutionary computational method to help the scheduling process of the Deep Space Network. Using a single objective approach, we were able to consistently obtain solutions that would improve the current best solution for three out of the four used objectives. We want to emphasize that our results are obtained in *one* run from a totally *random* population, whereas given DSN forecast weeks are obtained from a limited number of changes made in a mostly conflict-free schedule from the preceding week. Considering, the proximity of our solution to the ideal solution, in the last objective (minimizing the number of facility conflicts), our program already constitutes a potentially valuable tool for the DSN scheduling process. Moreover, with the performances on the first three objectives, we have a lot of leeway to make tradeoffs between objectives and therefore we will most likely be able to satisfy all four objectives used in this paper, when we implement a multi-objective approach. For this reason, we are optimistic that we will be able to propose a tool in the near future that will both improve the quality of the current schedule and generate alternative schedules meeting most of the requirements.

## REFERENCES


[1] C. Borden, Y.-F Wang, G. Fox, "Planning and Scheduling User Services for NASA's Deep Space Network," *NASA Planning and Scheduling Workshop*, 1997.

[2] Y.-F. Wang, A. Wax, R. Lam, J. Baldwin, and C. Borden, "Collaborative Scheduling Using JMS in a Mixed Java and .NET Environment," *IEEE International Space Mission Challenges for Information Technology (SMC-IT) Proceedings, pp. 505-512, July 2006.*

[3] Terrile, R. J., Adami, C., Aghazarian, H., Chau, S. N., Dang, V. T., Ferguson, M. I., Fink, W., Huntsberger, T. L., Klimeck, G., Kordon, M. A., Lee, S., von Allmen, P. A. and Xu, J. (2005) "Evolutionary Computation Technologies for Space Systems" IEEE Aerospace Conference Proceedings, Big Sky, MT, March 2005.

[4] Johnston, Mark D. (2006) "Multi-Objective Scheduling for NASA'S Future Deep Space Network Array", International Workshop on Planning and Scheduling In Space Proceeding, October 22-25, 2006.


## BIOGRAPHY

***Alexandre Guillaume*** *is a member of the information systems and computer science staff at the Jet Propulsion Laboratory. His recent work includes genetic algorithm application, quantum computing theory, experiments on superconducting devices and materials modeling. His work has been published in peer reviewed journals. He has a Ph. D. in Physics from the Joseph Fourier University in Grenoble (France).*

***Seungwon Lee*** *is a senior member of information systems and computer science staff at the Jet Propulsion Laboratory. Her recent work includes genetic algorithm application, high performance computing, materials modeling, and nonlinear dynamics system. Her work is documented in numerous journals and conference proceedings. She has a B.S. and M.S. in Physics from Seoul National University in Korea, and has a Ph. D in Physics from Ohio State University.*

***Robert Hovden*** *is an Academic Part Timer student employee at JPL, who is about to graduate from Georgia Tech, with a bachelors in Physics. He spent two summers working at JPL and his main focus is Genetic Algorithm related research. While at Georgia Tech he was employed as a math teaching assistant for two years. Robert's contribution to the scheduling and sequencing task include the implementation of SOAP messaging interface used in communicating with the Deep Space Network servers.*

***Hua Zheng*** *(AKA Will) is a software engineer in the Flight Avionics section at JPL. He has been involved in various research projects in the past related to wireless avionics, reconfigurable logic, fault-tolerant computing, and evolutionary computation. He is also a member of the Mar Science Laboratory avionics design team. His specialties include software development with GNU software, Linux device drivers, and FPGA development. Zheng has a BS degree in Computer Engineering from the University of California, Irvine, and is currently working on his MS*

*degree in Computer Science at California State University, Los Angeles.*

***Savio Chau** is a principal engineer and group supervisor at the Jet Propulsion Laboratory of the California Institute of Technology. He has 24 years of industrial experience in both research and development. He is currently the lead engineer of the Command and Data Handling Subsystem of the Jovian Icy Moon Orbiter project. He is also leading two technology development projects in the applications of Genetic Algorithm and the advanced data bus architecture for spacecraft. He has been a lecturer in the Computer Science Department at the University of California, Los Angeles. He current research interests include modular, high performance, intelligent, and ultra long life flight systems that utilize techniques from reconfigurable and evolvable hardware, neural network, genetic algorithms, fuzzy logic, and other biology-inspired techniques. He has 4 refereed journal publications, 20 conference papers, 2 patents, and 1 pending patent. He was the program co-chair of the 2002 Pacific Rim International Symposium on Dependable Computing (PRDC2002). He earned his BS in electrical engineering from Loyola Marymount University, Master of Engineering from California State Polytechnic University, Pomona, and Ph.D. in computer science from University of California, Los Angeles. He is a member of Tau Beta Pi and Eta Kappa Nu.*

***Yu-Wen Tung** is a senior member of JPL with more than 15 years of experience in mission software design, development and management. His research interests include modeling, simulation, parallel computing algorithms, software architecture, and model checking verification/validation techniques. He holds a Ph.D. degree in Electrical/Computer Engineering from the University of Southern California. Currently he is the group supervisor of the Modeling and Simulation for Planning and Execution Systems group in JPL.*

***Yeou-Fang Wang** is a senior member in the Planning and Execution Systems Section at the Jet Propulsion Laboratory. He has developed methodologies and tools for antenna load forecasting and scheduling systems for NASA's Deep Space Network. He has authored several journal and conference papers in the fields of artificial Neural Networks and scheduling. Dr. Wang's recent work includes automatic schedule generation algorithm development using computational intelligence, collaborative engineering, service oriented software architecture design, and multi-agent systems application. He has BS degree in Control Engineering from the National Chiao-Tung University of Taiwan and MS and PhD in Electrical and Computer Engineering from the University of California, Irvine.*

***Richard J. Terrile** created and directs the Center for Evolutionary Computation and Automated Design at NASA's Jet Propulsion Laboratory. His group has developed genetic algorithm based tools to improve on human design of space systems and has demonstrated that computer aided design tools can also be used for automated innovation and design of complex systems. He is a planetary astronomer and the co-discoverer of the Beta Pictoris circumstellar disk. Dr. Terrile has B.S. degrees in Physics and Astronomy from the State University of New York at Stony Brook and an M.S. and a Ph.D. in Planetary Science from the California Institute of Technology in 1978.*